IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

March 28, 2007

Serial No.	10/666,601
Applicant:	Hubert Lobo et al.
Filed:	09/18/2003
Title:	SYSTEM AND METHOD FOR ELECTRONIC SUBMISSION, PROCUREMENT, AND ACCESS TO HIGHLY VARIED MATERIAL PROPERTY DATA
Art Unit:	2167
Examiner:	Lovel, Kimberly.
Confirmation Number:	8543
Attorney Docket No.:	MA1-2

Commissioner for Patents
Alexandria, VA 22313-1450

PURPOSE OF DECLARATION

This declaration is to establish, under 37 C.F.R. §131 Applicant's invention of claims 1-4, 12-38, and 47 in U.S. Patent Application Ser. No. 10/666,601 prior to August 30, 2002, which is the priority date of U.S. PGPub 2005/0131861, cited in the Office Action of January 19, 2007, in support of the rejections under 35 U.S.C. § 103(a). This declaration is made by the first named inventor of the present application: Hubert Lobo.

FACTS AND DOCUMENTARY EVIDENCE

The undersigned inventor hereby declares as follows:

1. My name is Hubert Lobo, and I reside at 913 Wyckoff Rd. Ithaca, NY 14850

2. I am the joint inventor with Kurien Jacob of the invention disclosed in U.S. Patent Application Serial No. 10/666,601, filed September 18, 2003.

3. My co-inventor and I conceived the invention claimed in the present application in May 2002, which date is prior to **December 16, 2002**, the filing date of U.S. PGPub. 2004/0117397.

4. My co-inventor Kurien Jacob started implementation of the invention in June 2002, worked on it diligently thereafter, and it was reduced to practice when a first working version was produced in September 2002 and went into testing in October 2002.

5. A company was formed to develop and sell the product on September 27, 2002, and hardware and software necessary to deploy the system was acquired in December, 2002.

6. I hereby provide evidence showing conception of the invention coupled with due diligence from prior to the date of the reference to the reduction to practice on September 30, 2002 followed by test deployment on October 22, 2002. This evidence comprises:

a) Attached as Exhibit A is a copy of the original proposal dated 30th May 2002, describing the details of the claimed invention. These details are part of a confidential proposal and are not available publicly.

b) Attached as Exhibit B is a photograph dated August 2, 2002 of a planning board showing details of the implementation of the invention.

c) Attached as Exhibit C is a screen capture from the software development archive showing origination date and change history for the main web-page of the software.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Hubert Lobo

Date: APRIL 06, 2007

Proposal for development of an XML driven Data Repository

May 30, 2002
Kurien Jacob, Roots Computer Svcs, Basking Ridge, NJ

Executive Summary

In response to customer requests for reliable, real-time delivery of material test data in a variety of presentation formats, Datapoint Laboratories (referred to hereafter as the company), seeks to build a material test data repository. The data repository would serve structured test data, acquired through the in-house testing process or through established attributed sources. The data served would then be presented to the next data consumer. This would consist of, either the presentation logic for displaying the data to the customer, or popular CAE analytical packages operated by the customer, which could use the data as input for further processing. The data repository would support the primary business process of the company, whereby customer orders for tests or TestPaks™ are placed and then executed, while the test output results are maintained in the database under the ownership of the customer.

Existing Production Environment

In the existing environment, we have:

- A web-enabled customer database which houses customer information including order information. The order information includes the name of the test or Testpak™ to conduct, the information about the test sample and the customer billing information.
- The test output results, maintained as simple flat file structures. The test data is acquired automatically as a result of laboratory testing processes. The test results are converted into spreadsheets and then either printed out or stored as view only files for dispatch to the customer.
- A manual order fulfillment process.

Basic Functional Requirements

1. **Data Capture**
   All test data captured through the testing process must be maintained in the data repository. Furthermore, the system should allow for integration of test data from established third party sources such as material vendors.
2. **Retrieval**
   All captured test data must be stored for easy and quick retrieval, selective queries and with security restrictions on access.
3. **Flexibility**
   Provision must be provided, to accommodate an evolving test catalog, with each test being customizable to a limited extent by the customer.
4. **Reliability**
   The data capture, service and maintenance process must be reliable and predictable.

5. **Data Grading**

   The data acquired must be graded for reliability. This would depend on factors such as, whether the data is from a third party, and upon the reliability of information about test samples provided by the customer.

6. **Maintenance**

   The test data must be maintained, with changes only permitted to the access security level, data reliability factor, the data billing information and other information not related to the sample and its characterization. Further the data maintenance plan must take into account potential schema evolution.

7. **The Customer View**

   The customer must be able to:

   - Order new tests – Each order would be executed using industry standard online transactions.
   - Review/Cancel ordered tests.
   - Track the progress of a particular test order.
   - View results of the test data through standardized views of each test.
   - Print results using standardized templates for each test.
   - Download test data preformatted for input to CAE tools.
   - Query the repository through a limited set of test specific, material specific and manufacturer specific queries.
   - Purchase access to other test data – this option would be presented while querying the repository for data as stated above.
   - View historic test information that is owned by the customer

8. **All external views of the data must be based on a unified XML schema**

   The customer views of test data and queried test information must subscribe to an XML schema, which is closely related to the MATML schema. Furthermore, the unified view must include customer information such as preferences, billing information, etc.

9. **Current system reuse**

   The current system must be used as far as it can support the system's desired use-cases.

10. **Synchronization of current test output into the repository**

    Current test processes must be adapted to feed output data into the repository. Historic data must be loaded into the repository.

11. **A Use Case for Test Order and Fulfillment**

    In Fig. 1, a typical use case of the system is depicted, where the customer orders a test online. The administrator sets up the test to be offered in the test catalog, which would contain details about the test and the format of the result. The test is now made available to any registered user, who after browsing the test catalog, places an order for the test to be conducted. The online transaction would result in an update of the customer's records in the Customer Database and the initialization of the test information in the Test Database. The test is now conducted in the laboratory and the results are downloaded to the data repository. The customer can then view the test

results through the presentation logic, which would also be used by the administrator to verify the test result.

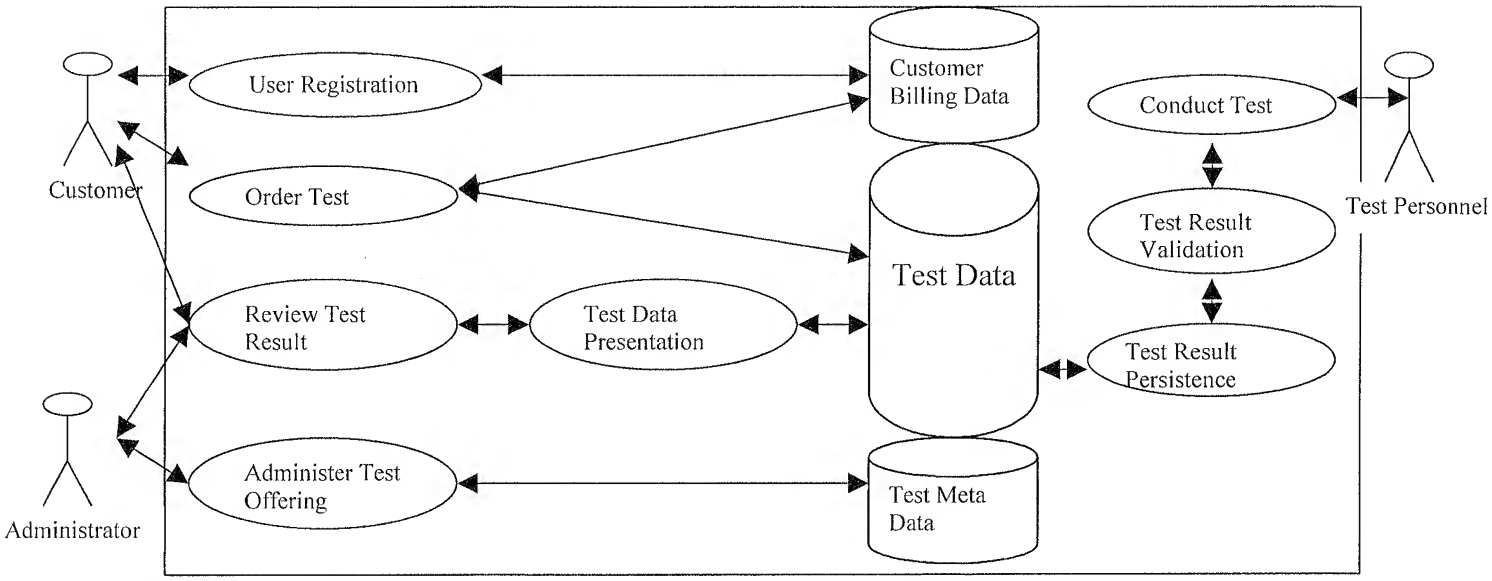


Fig 1. User Test Order and Fulfillment

1. **Choice of Data Store**

We face the choice of using a native RDBMS with middleware generating XML views of the data, or using native XML stores. In the former case we again face the following choices:

- Modeling the data purely as an XML schema, while leaving it up to the middleware to generate its internal mapping. Here we have the disadvantage that middleware may not optimally represent the data. This is a cause for concern, given the large volumes of test output data available.
- Modeling the data traditionally while managing the mapping to an XML schema. Here we face the task of managing the relational schema as well as the mapping from relational to XML schema. Care needs to be taken that the mapping layer is not overly complex or large, as the lifecycle management of such a layer may significantly add to the cost of the system.

In the case of using a native XML store, the current maturity of the technology is not comparable to that of relational stores in terms of managing the data. In addition there arises the question of sustainability of native XML DBMS packages in a market where large RDBMS vendors are entrenched.

2. **Data Modeling**

The volume of data output through the testing process is large. A typical type of test data is in the form of graphs. The question arises whether to store the graph data as structured data, keeping a record of each point value output. The use-cases (both immediate and future) for the output would determine the approach taken. If the graph data is always indivisibly exposed as a single graph, then the value of storing each point separately is minimal, in this case the graph can be stored as a single large object which modern database packages can efficiently support. However if there

would be a use-case where the user wishes to selectively query the graph data (e.g. requesting a plot of a subset of the parameters), then the expense of retrieving all graph data would be inordinately large, thus separating each point would be a preferable solution.

3. **Dynamic Schema**
   The suite of tests and TestPaks™ would be constantly changing. The tests offered have default values, which could change or be customized by the user. The test data format may differ depending on the context of the test (TestPak™ used). We must note that each change to the data schema would affect all processes feeding data into, and extracting data from, the system. Thus there would be a need to separate purely presentation related information from the data capture information, which would enable customizations of the presentation information without affecting the data capture information. In the case of using an XML-RDBMS hybrid solution, such a separation would also enable a generic presentation engine to be built, in order to expose the desired XML Schema. Such a presentation engine would work off the test *metadata* to build the output XML object.

Solution Framework

Architecture

1. The XML Schema

The XML Schema as proposed would be the customer view of all data in the database. The schema would be extended to include the existing customer information and billing information. This is required so that the presentation layer would be completely XML driven.

2. Proposed Solution Architecture

The proposed system would adopt the strategy of using an RDBMS store with an XML view being exposed through a presentation engine. Assuming a limited set of queries, which could be described as XPATH expressions, the complexity of such a presentation engine could be limited by utilizing existing RDBMS support for extracting XML documents from the database.

The test data that is captured would be passed through adaptors, which would feed the Test Database through efficient bulk data feeds. Here we would not require the test metadata to be submitted, resulting in a compact and efficient data upload.

The test metadata would share the lifecycle of the test offering in the catalog. The test metadata would now capture the presentation information and common test information. Administrative capability to edit and create and delete test types would be present.

The customer view would be driven off the XML schema, with high performance tasks such as graph rendering being pushed to the customer system.

The customers order and billing database would be integrated with the system to provide an extended XML schema, which composes the customer schema and the test data schema.

The system would be designed for 24x7 availability, with possible interruptions to service only while the test catalog is being updated.

3. Proposed High Level System Architecture

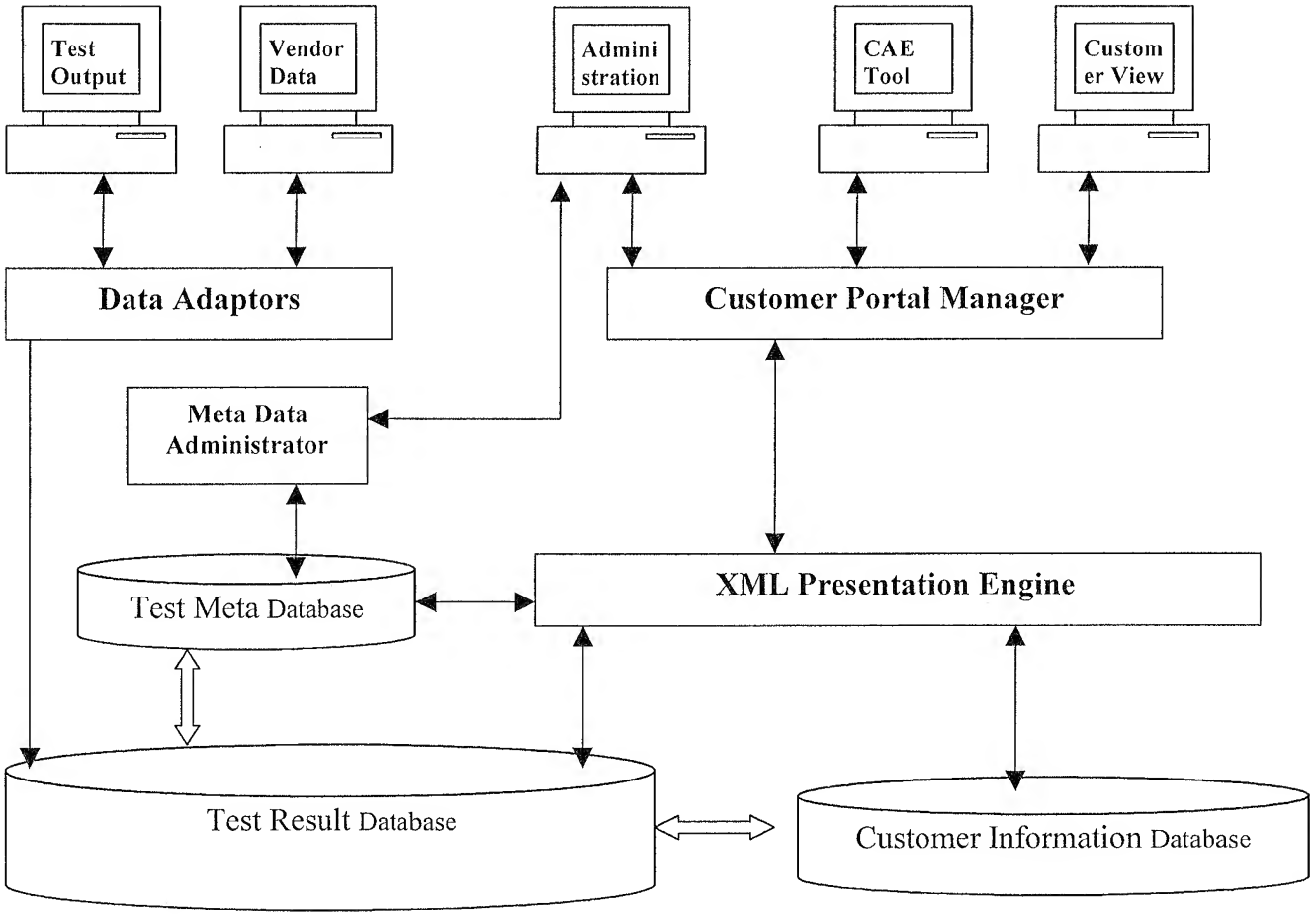


Fig. 2 System Components: Their interactions and relationships

4. **Maintenance of Test Meta Data**
   In order to accommodate continuous growth in the type of tests offered, there is a requirement to store test metadata. The test metadata would include the test catalogue data such as the test name, test type, test material characterization information, testing process description. In addition, the test metadata would include the description of test output data components. Thus both point data and the graphs captured in the output data would be described. Each component description would include the component's name/title, type (point/coefficient/graph) and the field descriptions for that component. Each field description would include the field name, the field's data type, acceptable value sets (ranges, enumerations), default values and the name of the field's physical unit (kg, in, J, etc). The administrator would be provided the capability to update the test catalog, i.e. add/edit tests, TestPaks™.
5. **Maintenance of Test Output Data**
   The test's data would be related to the metadata, which would describe it. The test's data would include generic test information, generic sample information and test component information. Standard test information would include data such as test instance identification, time and period of test, data ownership information, data access information and reliability information. Generic source sample information would include sample characterization information, manufacturing and post processing information. Test specific component information, as described by the metadata, would be maintained. All test data would identify the test type as specified by the test metadata.
6. **The Customer View**
   The customer would work in the context of a session where all requests and transactions are handled. The customer would have a portal from where repository data can be queried. Some of the functionality of this portal would be to:
   - Order new tests – Each order would be executed using industry standard online transactions.
   - Track the progress of a particular test order
   - View results of the test data through standardized views
   - Download test data preformatted for input to CAE tools.
   - Query the repository through a limited set of test specific, material specific and manufacturer specific queries.
   - Purchase access to other test data – this option would be presented while querying the repository for data as stated above.
7. **Synchronization of current test output into the repository**
   Existing test output processors would be adapted to feed the repository data that is consistent with the schema. This would involve validating the data and performing constraint checks as specified in the test metadata prior to feeding the repository. Historic test data, which is maintained as spreadsheet data, would have to be loaded into the database.

The Development Platform

The choices we have here are of development upon a commercial Java based platform, the Microsoft platform, or a proprietary platform. Given that the Java and Microsoft platforms effectively support our system architecture, the third option is discarded as being overly expensive. The Java based platform (referred to as the J2EE based Web Application framework), has large system vendor such as Sun Microsystems, IBM and BEA systems as well as smaller system vendors such as Macromedia. We have the advantage of portability and the choice of moving to a non-Microsoft based Operating Environment should the transaction processing requirements increase vastly. The Microsoft Platform (also known as the .NET platform) provides support for developing web-based services. In the current scenario, given that the existing customer database and order management system are built upon a Microsoft platform, it (the Microsoft supported platform) looks to be cost effective. Thus, in the current case the system would be developed using the Microsoft Windows operating platform, with the Microsoft SQL Server database manager, the Microsoft Internet Server and the .NET web-services platform. The hardware requirements for system development, apart from the current laboratory test process equipment, would include an Intel processor (preferably Xeon) based server, and client workstations for data presentation.

Project Stages

1. **Detailed Feature Matrix and phasing of the deliverables**
   During this stage the following activities would be completed:
   i. The detailed feature matrix of all system modules.
   ii. The phasing of delivery of all the system features.
   The deliverable for this stage is the detailed description of all use cases to be covered, the system feature matrix, the complete schema and the detailed project schedule for the full feature set (Approximate time span: 7 days).

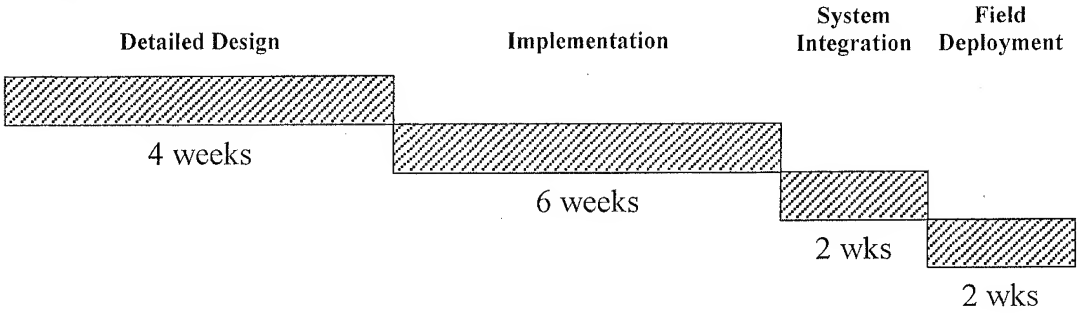
2. **Detailed Design of Phase One deliverables**
   In this stage, the detailed design of each system module would be undertaken. At the end of this stage, the detailed design of the system would be complete. The phasing of all the system features would be undertaken in this stage. The deliverable for this stage is the detailed design document. This would contain the detailed description of all components with their interfaces and the data structures used (Approximate time span: 20 days).
3. **Implementation of Phase One deliverables**
   At the end of this stage, the implementation and unit testing of all modules of the system identified as Phase One deliverables would be complete. The delivery for this stage is the documented code for each of the modules (Approximate time span: 20 days).
4. **System Integration and Testing of Phase One deliverables**
   At the end of this stage, the integration testing of all modules of the system would be complete. The deliverable for this stage is a field deployable system (Approximate time span: 15 days).
5. **Deployment and Field Testing for Phase One deliverables**

This stage would involve conducting customer trials of the system (Approximate time span: 15 days).

Milestones *(specific dates would depend upon feature matrix/start-date of project)*

1. XML Data Schema + Relational Mapping.
2. Feature Matrix + Detailed Schedule.
3. Design
   i. Design of XML Presentation Engine and Meta Data Administrator.
   ii. Design of Data Adaptors.
   iii. Design of Customer Portal Manager.
4. Implementation
   i. Implementation and Unit Testing of XML Presentation Engine and Meta Data Administrator.
   ii. Implementation and Unit Testing of Data Adaptors.
   iii. Design and Unit Testing of Customer Portal Manager.
5. System Integration.
6. Deployment and Field Testing.

Timeline *(detailed timeline would be based upon the Feature Matrix)*



Total Timeline for Phase 1: 16 weeks.

Delivery Plan

Each system module would be developed in isolation from the existing system, to avoid potential disruption of business processes. At the time of system integration the various modules would be deployed on a set of machines, which would reflect the final field deployment scenario. Deployment would be staged with the existing system running concurrently with the new system.

Project Costing

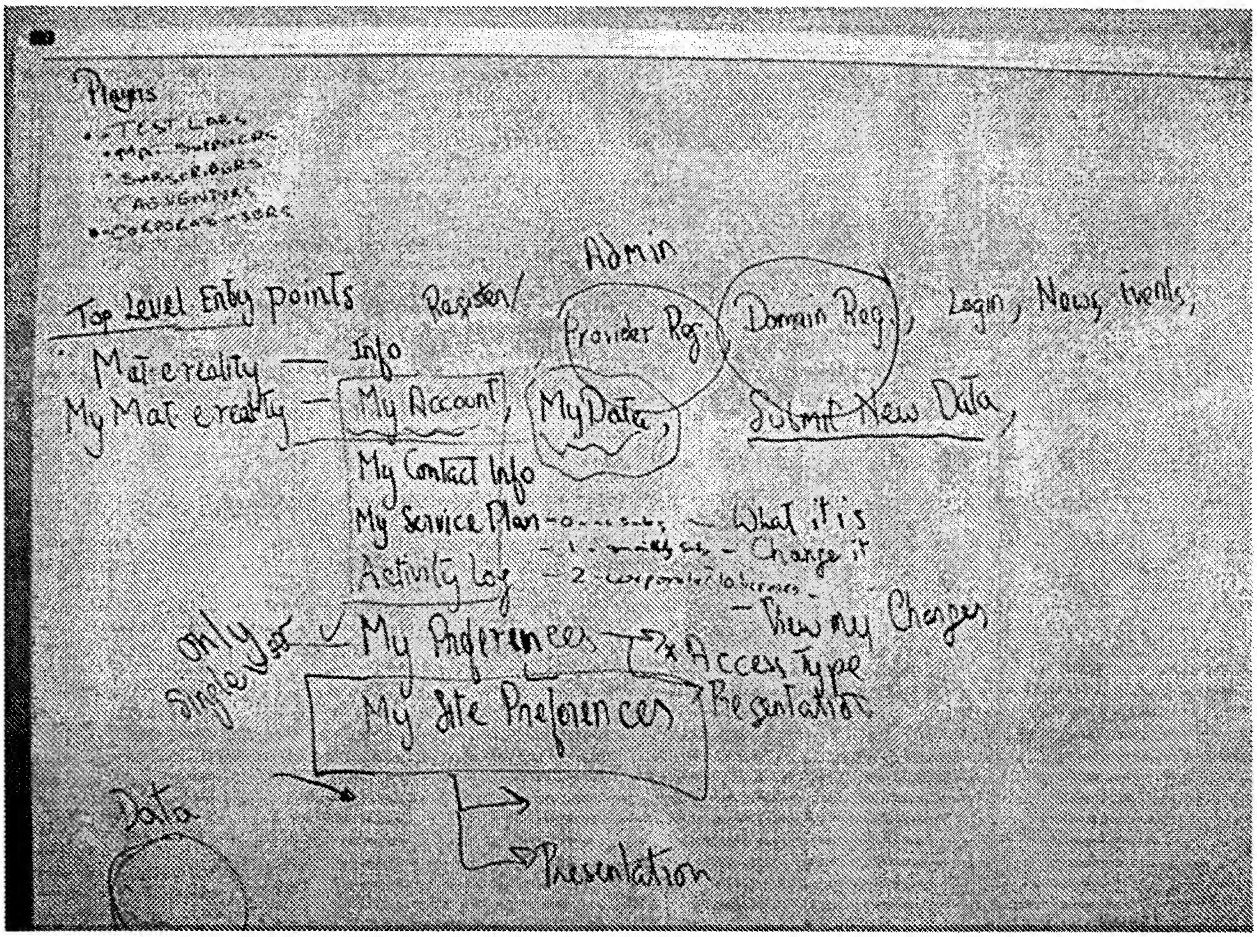
The cost of the project would involve:

1. Commercial software: Microsoft Server Software.
2. Hardware resources: 1 Server Workstation, 1 Client PC
3. Time and Money: Base rate of $60/hr (for a 40 hour week, no charges for extra hours).

Terms and Conditions *{as per mutual agreement}*

EXHIBIT B

Picture File DSC00186.JPG –, dated August 2, 2002 photograph of planning board, showing design and implementation of the system.



Proof of file creation date

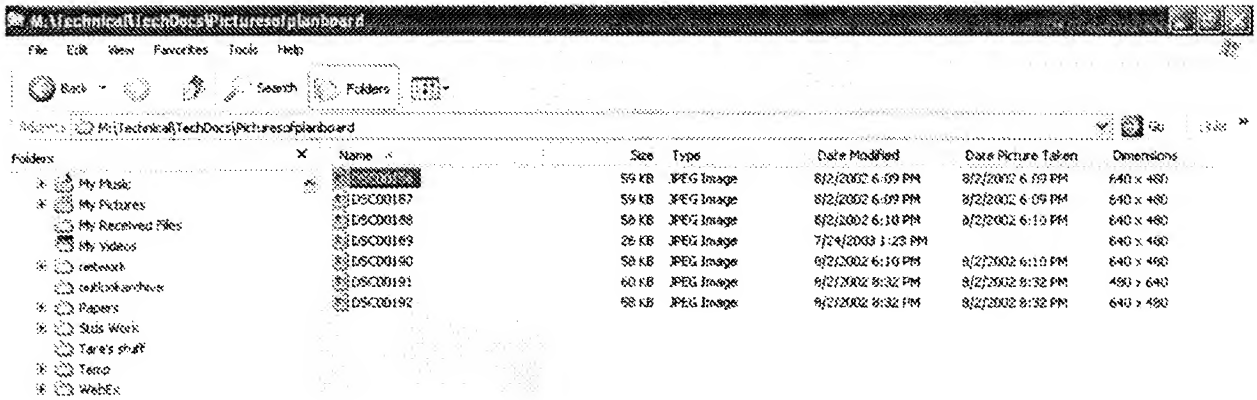


EXHIBIT C

Proof of Deployment with Change History for home page my.matereality/default.aspx.cs dated October 22, 2002. Record of other files related to software deployment showing October 22, 2002 creation date.